A Linear Programming Formulation of Flows over Time with Piecewise Constant Capacity and Transit Times

Juan Alonso and Kevin Fall

IRB-TR-03-007

June, 2003

DISCLAIMER: THIS DOCUMENT IS PROVIDED TO YOU "AS IS" WITH NO WARRANTIES WHATSOEVER, INCLUDING ANY WARRANTY OF MERCHANTABILITY NON-INFRINGEMENT, OR FITNESS FOR ANY PARTICULAR PURPOSE. INTEL AND THE AUTHORS OF THIS DOCUMENT DISCLAIM ALL LIABILITY, INCLUDING LIABILITY FOR INFRINGEMENT OF ANY PROPRIETARY RIGHTS, RELATING TO USE OR IMPLEMENTATION OF INFORMATION IN THIS DOCUMENT. THE PROVISION OF THIS DOCUMENT TO YOU DOES NOT PROVIDE YOU WITH ANY LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS

Copyright 2002, Intel Corporation, All rights reserved.

Intel Research Berkeley

A Linear Programming Formulation of Flows over Time with piecewise constant capacity and transit times

Juan Alonso[1] and Kevin Fall[2]

[1] Swedish Institute of Computer Science, Stockholm Sweden
[2] Intel Research, Berkeley USA

Abstract. We present an algorithm to solve a deterministic form of a routing problem in delay tolerant networking, in which contact possibilities are known in advance. The algorithm starts with a finite set of contacts with time-varying capacities and transit delays. The output is an optimal schedule assigning messages to edges and times, that respects message priority and minimizes the overall delivery delay. The algorithm consists of two main ingredients: a discretization step in which the raw data provided by the contacts is used to obtain appropriate subdivisions of the relevant time intervals, and a linear program, a dynamic version of the classical multicommodity flow problem, in which transit times are piecewise constant, and where both edges and nodes are capacitated (in the case of edges, with piecewise constant capacities). In fact, we present two equivalent LP formulations, of which one is smaller and runs faster in CPLEX, a general purpose linear solver.

1 Introduction

Route selecting in data networks typically consists of selection of shortest paths in a connected directed (or undirected) graph. In this work, we look at the problem of optimal message routing in a *delay tolerant network* (DTN)[1], which is assumed to suffer from frequent partitioning such that no contemporaneous end-to-end path may ever exist. In graph theoretic terms, this problem is a form of dynamic multicommodity flow or quickest transshipment problem in which both edge capacities and transit delays along an edge can vary as a function of time and are permitted to be zero, and finite buffer capacities are assigned to each vertex. Examples of real networks exhibiting such behavior include Low-Earth Orbiting Satellites, which provide network connectivity for predictable intervals of time (and at predictable capacities) based on orbital mechanics.

For purposes of this study, we assume that a set of vertices V and a set of *contacts* are given. A contact C is a directed edge between two vertices in V for a particular interval of time in which a constant capacity and latency is available to carry commodity flow. More precisely, a contact C is a 4-tuple (e, I, c, d), where $e \in V \times V$, I is a time interval during which flow is permitted to *depart* along edge e, and c, d are non-negative real numbers representing edge capacity and transit time, respectively. In practice, the time intervals are finite for contacts describing links (communication opportunities) that come and go periodically or occasionally, and are $(-\infty, \infty)$ for persistent ones (persistent links are thus assumed to have a perpetual constant capacity and delay). Given the set of vertices and contacts, we construct the routing (multi)graph $G = (V, E)$ where $E = \{e | e$ is an edge in any contact given$\}$.

In addition to the topology elements of vertices and contacts, we assume that each vertex also has an associated maximum buffer capacity b_v, and that there is a commodity demand matrix M known in advance where $M = \{m_{k,p} | k = (u, v) \in V \times V, p \in (1 \dots 4)\}$, where p is a priority level. In addition, we assume a weight function $w(p)$ to be given that gives the relative penalty paid for delaying high-priority traffic. Its use is explained below.

Our approach is to use a linear programming formulation of this problem, as it is perhaps the most natural mechanism to express the various constraints. Our goal is to capture the essence of the problem as closely as possible, with less emphasis on proof of polynomial-time algorithm operation. Future work will pursue the computational complexity of this problem, and of related problems in which M or E are not deterministically known with certainty in advance.

2 A system of time intervals

As mentioned above, the model requires special care with the time intervals used. In the LP formulation of the problem given in Section 3, it is assumed that we are given a system of time intervals and subdivision points satisfying a certain property (see condition (i) below). In this section we explain how to obtain (some of) these subdivision points from first principles (i.e. from the *contacts* described above). In Sections 4 and 5, we complete the algorithm and give proofs that the subdivision points so constructed do satisfy the required condition.

We assume that G is constructed as described above and that "reality" has provided us with a finite set $\mathcal{C} = \{C_1, \ldots, C_n\}$ of contacts. Here "reality" also means that when the delay (and/or capacity) of a certain link is not constant over a given time interval, we subdivide the time interval into subintervals so that the variable delay can be satisfactorily approximated by constant delays over the subintervals. This situation will thus be modelled as a finite number of contacts of the type defined above.

By collecting the endpoints of all *finite* time intervals of contacts in $\mathcal{C}$, we form a set T_1 of time-points, i.e.[3]

$$T_1 = \{t | \exists C = (e, I, c, d) \in \mathcal{C} \text{ with } I \text{ finite, such that } t = \iota(I) \text{ or } t = \tau(I)\}.$$

We can think of T_1 as a subdivision of the *transmission* time interval $[t_0, t_h]$, where $t_0 = \min T_1$, and $t_h = \max T_1$. Collecting the delays of all contacts, we form the set $D = \{d | d \text{ is the delay of some } C \in \mathcal{C}\}$, and set $\Delta = \max D$. We also have a *reception* interval $[t_0, t_h + \Delta]$ and an *extended* interval $[t_0 - \Delta, t_h + \Delta]$, see Fig. 1. Observe that transmission on any link occurs entirely in time-interval $[t_0, t_h]$ and that, due to delay, transmissions are (completely) received in the period $[t_0, t_h + \Delta]$ (notice that delay can be zero in some links).

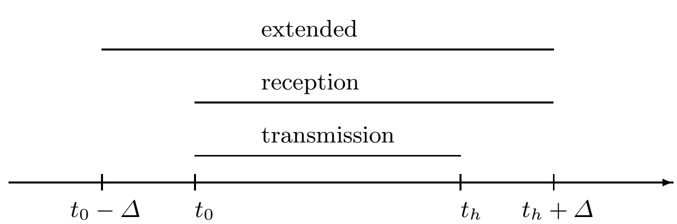

Fig. 1. Time intervals.

We extend T_1 to $T_2 = T_1 \cup \{t_0 - \Delta, t_h + \Delta\}$. Observe that T_2 is a subdivision of the extended interval, $T_2 \cap [t_0, t_h + \Delta]$ is a subdivision of the reception interval, and $T_1 = T_2 \cap [t_0, t_h]$ is a

[3] Here, for any ordered pair $X = (a, b)$, $\iota(X) = a$ and $\tau(X) = b$.

subdivision of the transmission interval. At every subdivision, T_2 generates a set of subintervals (of the extended interval), denoted $\mathcal{I}_2$, as follows. If $T_2 = \{t_0 - \Delta, t_0, t_1, \ldots, t_h, t_h + \Delta\}$, then $\mathcal{I}_2 = \{[t_0 - \Delta, t_0], [t_0, t_1], \ldots, [t_h, t_h + \Delta]\}$.

Next, we define a capacity function $c : E \times \mathcal{I}_2 \to \mathbf{R}_+ \cup \{0\}$ [resp. delay function $d : E \times \mathcal{I}_2 \to \mathbf{R}_+ \cup \{0\}$]. The set of contacts satisfy constraints that imply that, given a pair $(e, I) \in E \times \mathcal{I}_2$, only one of the following possibilities can occur: there is a unique contact $C' = (e', I', c', d') \in \mathcal{C}$ with $e = e'$ and $I \subseteq I'$, or else no contact involving e has a time interval containing I. In the first case we define $c_{e,I} = c'$ [resp. $d_{e,I} = d'$], and in the second case $c_{e,I} = 0$ [resp. $d_{e,I} = 0$].

At this point we have constructed a time axis and identified transmission, reception and extended intervals which are provided with compatible subdivisions. To proceed further and define the required property, we need to further subdivide the constructed intervals. More exactly, we need to refine the given subdivisions T_1 and T_2. The precise details of the construction and proofs are given later, in sections 4 and 5 below. For the time being, we content ourselves with making precise the assumptions needed for the LP formulation.

Thus, assume we are given a finite subdivision T_E of the extended interval $[t_0 - \Delta, t_h + \Delta]$. We will assume, moreover, that T_E contains t_0 and t_h. This condition ensures that $T_X = T_E \cap [t_0, t_h]$ and $T_R = T_E \cap [t_0, t_h + \Delta]$ are (finite) subdivisions of the transmission interval $[t_0, t_h]$ and of the reception interval $[t_0, t_h + \Delta]$, respectively.

From these subdivisions we generate subintervals in the usual way: order the elements of, say, T_E and let $\mathcal{I}_E$ consist of all intervals of the form $[t_i, t_{i+1}]$, where t_i, t_{i+1} are consecutive elements of T_E. Observe that $|\mathcal{I}_E| = |T_E| - 1$, the intersection of any pair of different subintervals of $\mathcal{I}_E$ is either a point of empty, and the union of all of them is the extended interval $[t_0 - \Delta, t_h + \Delta]$. Similarly, T_X [resp. T_R] defines a set of subintervals $\mathcal{I}_X$ [resp. $\mathcal{I}_R$] whose union is the transmission [resp. reception] interval. Observe that $\mathcal{I}_X \subseteq \mathcal{I}_R \subseteq \mathcal{I}_E$.

For an interval $I = [a, b]$ and real number d, we let $I \pm d$ denote the interval $[a \pm d, b \pm d]$. The condition mentioned above can be expressed as follows:

i) $\quad \forall d \in D \qquad \forall I \in \mathcal{I}_R, \qquad I - d \in \mathcal{I}_E$

In the input to the problem formulated below we assume that this condition is satisfied.

3 LP formulation

In this section we describe a linear programming model for the routing problem described in Section 1. The model is a dynamic version of the classical multicommodity flow problem. The commodities are source-destination pairs. Another appropriate choice of commodities would have been triples of the form source-destination-priority. We model delay by distinguishing between transmission and reception, i.e. we introduce two parallel sets of variables, one set to take care of transmission (transmission variables are denoted X) and the other to take care of reception (reception variables are denoted R). In the model, we consider transmission to be the primal activity and reception the derived one. We begin by giving a list of the input to the model.

Sets and Functions

G directed graph, $G = (V, E)$

T_E finite subdivision of $[t_0 - \Delta, t_h + \Delta]$

T_X finite subdivision of $[t_0, t_h]$ induced by T_E, $T_X = T_E \cap [t_0, t_h]$

T_R finite subdivision of $[t_0, t_h + \Delta]$
induced by T_E, $T_R = T_E \cap [t_0, t_h + \Delta]$
$\mathcal{I}_E$ set of time intervals induced by T_E
$\mathcal{I}_X$ set of time intervals induced by T_X
$\mathcal{I}_R$ set of time intervals induced by T_R
c $c : E \times \mathcal{I}_E \to \mathbf{R}_+ \cup \{0\}$, where $c_{e,I}$ is the (transmission) capacity of edge e on time interval $I \in \mathcal{I}_E$, and $c_{e,I} = 0$ for all e, whenever $I \subseteq [t_0 - \Delta, t_0] \cup [t_h + \Delta]$
d $d : E \times \mathcal{I}_E \to \mathbf{R}_+ \cup \{0\}$, where $d_{e,I}$ is the propagation delay of edge e during time interval $I \in \mathcal{I}_E$,
and $d_{e,I} = 0$ for all e, whenever $I \subseteq [t_0 - \Delta, t_0] \cup [t_h + \Delta]$
K set of commodities, $K \subseteq V \times V$
P set of priorities, $P = \{1, 2, 3, 4\}$
w weight function $w : P \to \mathbf{R}_+$,
strictly monotone decreasing
I^v set of incoming edges (arc heads) at node v
O^v set of outgoing edges (arc tails) at node v

Constants
Δ largest delay, $\Delta = \max D$
b_v buffer capacity of node v
$m_{k,\ell}$ size of all messages from $\iota(k)$ to $\tau(k)$ which have priority ℓ

Variables
$N^{k,\ell}_{v,t}$ amount of commodity k of priority ℓ occupying the buffer at node v at time $t \in T_R$
$X^{k,\ell}_{e,I}$ amount of data from $\iota(k)$ to $\tau(k)$ and priority ℓ that is *transmitted* at $\iota(e)$ (sent to $\tau(e)$) during $I \in \mathcal{I}_E$
$R^{k,\ell}_{e,I}$ amount of data from $\iota(k)$ to $\tau(k)$ and priority ℓ that is *received* at $\tau(e)$ (coming from $\iota(e)$) during $I \in \mathcal{I}_R$

When $e = (v, w)$ is a link from node v to node w, we let $\iota(e) = v$ denote the initial point (arc tail) of e, and $\tau(e) = w$ denote its terminal point (arc head). Note that $X^{k,\ell}_{e,I}$ has been defined for $I \in \mathcal{I}_E$, i.e. even for I outside the transmission interval. This does not contradict the fact that transmission occurs only for $I \in \mathcal{I}_X$ since it will turn out (from equations (5) below, together with the conditions on c imposed above) that $X^{k,\ell}_{e,I} = 0$ for all $I \subseteq [t_0 - \Delta, t_0] \cup [t_h + \Delta]$. Observe once again that c stands for the capacity of an edge *at its tail* (the departing point of a link). It is an implicit assumption of the model that the receiver at the far end of the link is capable of receiving at the appropriate times so that all transmissions (occurring at the initial point of the link) over $[t_0, t_h]$ will be received (at the end point of the link) over $[t_0, t_h + \Delta]$.
In formulas (1) (2) below we need to relate time intervals to their endpoints. We do this by assuming the time intervals are numbered $I_1, \ldots, I_q, \ldots$, and satisfy $I_q = [t_{q-1}, t_q]$.

LP formulation

$$\min \sum_{\ell \in P} \sum_{k \in K} \sum_{e \in E} \sum_{I_q \in \mathcal{I}_X} w(\ell) \cdot t_q \cdot X^{k,\ell}_{e,I_q} \tag{1}$$

subject to

$$\sum_{w\in I^v} R^{k,\ell}_{(w,v),I_q} - \sum_{w\in O^v} X^{k,\ell}_{(v,w),I_q} = N^{k,\ell}_{v,t_q} - N^{k,\ell}_{v,t_{q-1}}, \quad k,\ell,v,I_q \in \mathcal{I}_R \tag{2}$$

$$R^{k,\ell}_{e,I} = X^{k,\ell}_{e,(I-d_{e,I})}, \qquad k,\ell,e,I \in \mathcal{I}_E \tag{3}$$

$$\sum_{\ell\in P}\sum_{k\in K} N^{k,\ell}_{v,t} \le b_v, \qquad k,\ell,v,t \in T_R \tag{4}$$

$$\sum_{\ell\in P}\sum_{k\in K} X^{k,\ell}_{e,I} \le c_{e,I}\cdot |I|, \qquad k,\ell,e,I \in \mathcal{I}_E \tag{5}$$

$$\sum_{I\in\mathcal{I}_R}\sum_{w\in O^v} X^{k,\ell}_{(v,w),I} - \sum_{I\in\mathcal{I}_R}\sum_{w\in I^v} R^{k,\ell}_{(w,v),I} \quad \begin{cases} 0 & \text{if } v\neq\iota(k)\ \&\ v\neq\tau(k) \\ m_{k,\ell} & \text{if } v=\iota(k) \\ -m_{k,\ell} & \text{if } v=\tau(k) \end{cases} \tag{6}$$

$$N^{k,\ell}_{v,t_0} = \begin{cases} m_{k,\ell} & \text{if } v=\iota(k) \\ 0 & \text{if } v\neq\iota(k) \end{cases}, \quad k,\ell,v \tag{7}$$

There is an equation of the form (6) for all $v\in V$, $k\in K$, $\ell\in P$. The notation $k,\ell,v,I_q\in\mathcal{I}_R$ in (2) means that there is an equation of this form for all $k\in K$, $\ell\in P$, $v\in V$ and $I_q\in\mathcal{I}_R$, and similarly for the other equations and inequalities.

Minimizing the objective function (1) favors transferring data as early as possible. The weight function w can be constructed to assign a large value to messages of high priority, thus making it very expensive to delay these messages. The left-hand side of equation set (2) is the difference between total incoming and total outgoing flow at a node, over a time-interval, for a commodity with given priority, whereas the right-hand side gives the net buffer occupancy, for the same node, commodity and priority, at the end of the same time-interval. Equations (3) state that the traffic received at the end point of e during time interval I is equal to the traffic transmitted at the initial point of e during $I-d_{e,I}$. In order for equations (3) to make sense, we need to guarantee that $X^{k,\ell}_{e,(I-d_{e,I})}$ is one of the variables *in* the model, i.e. that $I-d_{e,I}\in\mathcal{I}_E$. This is precisely what condition (i) says. (4) expresses the fact that the total buffer occupancy at a node can never exceed the buffer capacity at the node, (5) that the total flow over a link during a certain time-interval cannot exceed the link capacity multiplied by the amount of time the link is open. The left hand-side of (6) computes the final (i.e. when *all* time has elapsed) difference between outgoing and incoming flow of a commodity k with priority ℓ, at a vertex v; the right-hand side says that this is equal to zero at transient nodes, and equal to $m_{k,\ell}$ [resp. $-m_{k,\ell}$] when $v=\iota(k)$ [resp. $v=\tau(k)$]. Finally, equations (7) are initial conditions expressing the fact that, at the beginning, only those nodes having messages to send have an occupied buffer, in the expressed amount.

4 Algorithm to obtain the time intervals

In this section we list the steps of an algorithm that starts from contacts and produces time intervals satisfying (i). Observe that the output of the algorithm is part of the input needed to formulate the linear program of Section 3.

0) `Start with a set of contacts and vertices, and construct the directed graph` $G = (V, E)$.
1) `Extract the time points to form` T_1 `and the delays to form` D. `Set` $\Delta = \max D$.
2) `Set` $t_0 = \min T_1$, $t_h = \max T_1$. `Set` $T_2 = T_1 \cup \{t_0 - \Delta, t_h + \Delta\}$.
3) `From` T_2 `construct the induced set` $\mathcal{I}_2$ `of subintervals.`
4) `Use the contacts to define the capacity and delay functions` $c, d : E \times \mathcal{I}_2 \to \mathbf{R} + \cup\{0\}$.
5) `Form the set` $D' = D \cup \{|t_i - t_{i-1}| : t_{i-1}, t_i$ `are consecutive elements of` $T_2\}$ `and denote its elements` $D' = \{d_1, \ldots, d_N\}$.
6) `Approximate the numbers in` D' `by rational numbers. From now on, we assume the elements of` D' `have the form` $d_i = n_i/m_i$, `for appropriate integers` n_i, m_i.
7) `Compute` $q = \mathrm{lcm}\{m_1, \ldots, m_N\}$, $p = \gcd\{n_1 \cdot (q/m_1), \ldots, n_N \cdot (q/m_N)\}$, `and set` $\epsilon = p/q$.
8) `Let` $T_\epsilon = \{t_0 + \ell \cdot \epsilon | \ell \in \mathbf{Z}\}$. `Then the subdivisions` $T_E = T_\epsilon \cap [t_0 - \Delta, t_h + \Delta]$, $T_R = T_\epsilon \cap [t_0, t_h + \Delta]$, `and` $T_X = T_\epsilon \cap [t_0, t_h]$ `of, respectively, the extended, reception and transmission intervals, together with their induced set of subintervals` $\mathcal{I}_E, \mathcal{I}_R$, `and` $\mathcal{I}_E$ `satisfy condition (i).`
9) `Use 4) to define capacity and delay functions` $c, d : E \times \mathcal{I}_E \to \mathbf{R} + \cup\{0\}$.

Steps 0)-4) were explained in section 2. Steps 5)-8) are discussed in the next section. For step 9), use the fact that every interval in $\mathcal{I}_E$ is contained in a unique interval in $\mathcal{I}_2$, together with the functions defined in step 4), to extend these definitions to $E \times \mathcal{I}_E$.

5 Proofs

The basic idea behind steps 5)-8) is to subdivide the set of time points so that the corresponding time intervals have the property that, when translated by a delay (i.e. going from I to $I + d_{e,I}$), they remain time intervals. In this section we give a recipe to do this, show that it is efficient, and prove that the resulting time intervals have the stated property. The crucial step to achieve this is the construction of ϵ, a positive rational number with the property that every delay and every length of an interval in $\mathcal{I}_2$ is an integral multiple of ϵ (cfr. Corollary 1 below).

Suppose we are given sets $T = \{t_0, \ldots, t_h\}$ and $D = \{d\}$ as above. Note that $T = T_1$ of the previous sections; we use T here to simplify the notation. We enlarge D by adding the lengths of the subintervals defined by T_1, thus $D' = D \cup \{t_1 - t_0, \ldots, t_h - t_{h-1}\}$. We assume the elements of D' to be rational numbers, say $D' = \{d_1, \ldots, d_N\}$ with $d_i = n_i/m_i$ $(n_i, m_i \in \mathbf{Z})$. Furthermore, we assume that the fractions cannot be simplified and that 0, if it is an element of D', is represented as 0/1. Let

$$q = \mathrm{lcm}(m_1, \ldots, m_N), \qquad p = \gcd(n_1 \cdot (q/m_1), \ldots, n_N \cdot (q/m_N))$$

and set $\epsilon = \frac{p}{q}$. Note that ϵ is a positive rational number. The main property of this number is that any integral linear combination of the d_i is an integral multiple of ϵ. This can be expressed more formally as follows. Let $\mathbf{Z}d = \{\ell \cdot d | \ell \in \mathbf{Z}\}$ denote the set of integral multiples of d, and let $A + B = \{a + b | a \in A,\ b \in B\}$ denote the set of sums of elements of A and B. With this notation we have:

Lemma 1.

$$\mathbf{Z}d_1 + \cdots + \mathbf{Z}d_N = \mathbf{Z}\epsilon$$

Proof. Note that p is defined as the greatest common divisor of a finite number of integers. By well-known properties of the *gcd*, it follows that

$$\mathbf{Z}(n_1 \cdot (q/m_1)) + \cdots + \mathbf{Z}(n_N \cdot (q/m_N)) = \mathbf{Z}p \tag{8}$$

To prove $\mathbf{Z}\epsilon \subseteq \mathbf{Z}d_1 + \cdots + \mathbf{Z}d_N$ it suffices to show that ϵ can be expressed as an integral linear combination of the d_i. By (8),

$$p = \ell_1 \cdot n_1 \cdot (q/m_1) + \cdots + \ell_N \cdot n_N \cdot (q/m_N)$$

for appropriate integers $\ell_1, \ldots, \ell_N$. Dividing by q,

$$\epsilon = p/q = \ell_1 \cdot d_1 + \cdots + \ell_N \cdot d_N$$

as desired. The reversed inclusion is proved similarly. This concludes the proof.

Corollary 1. *ϵ divides every $d_i \in D'$, i.e. $d_i/\epsilon \in \mathbf{Z}$.*

Proof. Since $d_i = 0 \cdot d_1 + \cdots + 1 \cdot d_i + \cdots + 0 \cdot d_N$ is an integral linear combination of the elements of D', Lemma 1 implies $d_i = r \cdot \epsilon$ for some integer r, as desired.

Given T and ϵ as above, let $T_\epsilon = \{t_0 + \ell \cdot \epsilon | \ell \in \mathbf{Z}\}$. Note that, despite the notation, T_ϵ is actually defined by both T and D.

Lemma 2. *T is contained in T_ϵ, i.e. $T \subseteq T_\epsilon$.*

Proof. The lemma follows easily from Corollary 1. For instance, since we can find an integer r such that $t_1 - t_0 = r \cdot \epsilon$, it follows that $t_1 = t_0 + r \cdot \epsilon$ is in T_ϵ. The proof can be completed by induction.

In the next result we use the notation $A \pm b = \{a \pm b | a \in A\}$.

Lemma 3. *For all $d \in D'$, $T_\epsilon \pm d = T_\epsilon$.*

Proof. We show that $T_\epsilon \subseteq T_\epsilon + d$. By Corollary 1, there is an integer r such that $d = r \cdot \epsilon$. Then $t_0 + \ell \cdot \epsilon = t_0 + (\ell - r) \cdot \epsilon + d \in T_\epsilon + d$, since $(\ell - r)$ is an integer. The reversed inclusion is proved similarly. Similarly for $-$ instead of $+$.

Lemma 4. *Suppose that $Y \subseteq \mathbf{R}$ is a subset of real numbers that satisfies:*

i) $T \subseteq Y$
ii) *for all $d \in D'$, $Y + d = Y$*

then $T_\epsilon \subseteq Y$.

Proof. Observe that condition $Y + d = Y$ is equivalent to $Y - d = Y$. Next, for any $A \subseteq \mathbf{R}$ we construct inductively a set $\hat{D}(A)$ as follows. Set $D^0(A) = A$, $D^1(A) = \{a \pm d | a \in A,\ d \in D'\}$, $D^n(A) = D^1(D^{n-1}(A))$ for $n \geq 2$, and define

$$\hat{D}(A) = \bigcup_{n \geq 0} D^n(A).$$

Note that if $A \subseteq Y$, then $D^1(A) \subseteq Y$ (by (ii) and the observation at the beginning of the proof). This, together with the fact that both $D^0(T)$ and $D^1(T)$ are subsets of Y, imply that $\hat{D}(T) \subseteq Y$.

We complete the proof by showing that $T_\epsilon \subseteq \hat{D}(T)$. Suppose that $\epsilon = \ell'_1 \cdot d_1 + \cdots + \ell'_N \cdot d_N$, by Lemma 1. Then a typical element of T_ϵ has the form $t_0 + r \cdot \epsilon = t_0 + \ell_1 \cdot d_1 + \cdots + \ell_N \cdot d_N$, where $\ell_i = r \cdot \ell'_i$. But $t_0 + \ell_1 \cdot d_1 \in D^{\ell_1}(T)$, $(t_0 + \ell_1 \cdot d_1) + \ell_2 \cdot d_2 \in D^{\ell_2}(D^{\ell_1}(T)) = D^{\ell_1 + \ell_2}(T)$, and so on. We conclude that $t_0 + r \cdot \epsilon \in D^{\ell_1 + \cdots + \ell_N}(T) \subseteq \hat{D}(T)$, as desired (a similar argument shows that T_ϵ is actually equal to $\hat{D}(T)$, but we don't need this fact). This concludes the proof.

Theorem 1 below shows that the constructed set T_ϵ is efficient, in the sense that it is contained in any other set satisfying similar conditions.

Theorem 1. *T_ϵ is the smallest set of numbers that contains T and is closed under translation by elements of D'.*

Proof. Let Y denote the smallest set that contains T and is closed under translation by D'. By Lemma 4, $T_\epsilon \subseteq Y$. By Lemma 2 and 3, T_ϵ contains T and is closed under translation by D'. Since Y is the smallest such set, $Y \subseteq T_\epsilon$, as desired.

Next, let $T_E = T_\epsilon \cap [t_0 - \Delta, t_h + \Delta]$, $T_R = T_\epsilon \cap [t_0, t_h + \Delta]$, and $T_X = T_\epsilon \cap [t_0, t_h]$, together with their induced set of subintervals $\mathcal{I}_E, \mathcal{I}_R,$ and $\mathcal{I}_E$ be as in 9) of the previous section. We will show that condition (i) is satisfied.

Theorem 2. *Condition (i) of section 2 is satisfied, i.e. for all $d \in D$ and $I \in \mathcal{I}_R$, we have $I - d \in \mathcal{I}_E$.*

Proof. Consider an interval of the form

$$I_r = [t_0 + r \cdot \epsilon, t_0 + (r+1) \cdot \epsilon]$$

for some integer r. Lemma 3 implies that, for all $d \in D'$, $I_r \pm d = I_s$ for some integer s. In other words, translation by elements of D' permutes the set of intervals of the form I_r. To complete the proof of (i) it remains only to check that if I_r is in $\mathcal{I}_R$ (i.e. if $I_r \subseteq [t_0, t_h + \Delta]$), then $I_r - d \in \mathcal{I}_E$ (i.e. $I_r - d \subseteq [t_0 - \Delta, t_h + \Delta]$). This is trivially true.

6 An equivalent LP formulation

In this section we present an equivalent formulation of the program in Section 3. A possible advantage of the new formulation is that the program itself, even though the number of constraints is unchanged, becomes smaller because the new constraints can be expressed more succinctly. We have tried the two formulations on CPLEX 7.1 (with default settings). The text file for the tighter formulation is, in our example, about 20% smaller and runs about 10% faster than the formulation in section 3.

To derive the new formulation, notice that, up to sign, the left-hand side of each equation of type (6), denoted $LHS(6)$, equals the sum over $I \in \mathcal{I}_R$ of the left-hand side of the corresponding equation of type (2), denote $LHS(2)$. On the other hand, the sum over $I \in \mathcal{I}_R$ of the right-hand side of an equation of type (2), denoted $RHS(2)$ is a "telescopic" sum that simplifies, collapsing to the difference of the last and initial terms. In symbols (where, in order to simplify the notation, we let $t_f = t_h + \Delta$ denote the last time point, and t_{f-1} denote the previous one),

$$\begin{aligned} -LHS(6) &= \sum_{I \in \mathcal{I}_R} LHS(2) \sum_{I \in \mathcal{I}_R} (N^{k,\ell}_{v,t_q} - N^{k,\ell}_{v,t_{q-1}}) && (9) \\ &= N^{k,\ell}_{v,t_1} - N^{k,\ell}_{v,t_0} + N^{k,\ell}_{v,t_2} - N^{k,\ell}_{v,t_1} + \cdots + N^{k,\ell}_{v,t_f} - N^{k,\ell}_{v,t_{f-1}} && (10) \\ &= N^{k,\ell}_{v,t_f} - N^{k,\ell}_{v,t_0} = -RHS(6) && (11) \end{aligned}$$

Finally, from equations (7) and (11) we obtain:

$$N_{v,t_f}^{k,\ell} = \begin{cases} m_{k,\ell} & \text{if } v = \tau(k) \\ 0 & \text{if } v \neq \tau(k) \end{cases} \tag{12}$$

We have thus shown that the "old" polyhedron, defined by the constraints (2)-(7), is contained in the polyhedron defined by the constraints (2)-(5) together with the constraints (7) and (12). However,it is easy to see that the converse is also true: the two polyhedra coincide. Thus, the new program: minimize the objective function (1) subject to equations (2), (3), (4), (5), (12) and (7), is equivalent to the program in section 3.

The interpretation of the set of equations (12) is clear: it requires that, at the final time, only the node $\tau(k)$ will contain commodity k (of any priority) in the amount $m_{k,\ell}$, while all other nodes will be empty of this commodity (any priority).

7 Related Work

The formulation of max flow for networks with constant positive capacities and arc traversal times originates with Ford and Fulkerson[4, 5], where a *time-expanded* graph is used to capture both arc capacities as well as transit delays. Variations and extensions of this approach have been used extensively in approaches to several *flows-over-time* problems including maximum flow, transshipment, and others. The attraction of time-expanded graphs arises due to the ability to take advantage of the rich set of algorithms for static graphs that can be applied to the time expanded graph to solve flows-over-time problems. The primary disadvantage of these approaches relates to the non-polynomial growth in graph size when constructing the time-expanded graph, which is generally expanded as a function of the time discretization selected. By using coarser discretizations, efficient approximation algorithms have been developed. Fleischer and Skutella construct a *condensed* time expanded graph[7] which can improve the time-expanded graph size down to a polynomial of the original graph and can potentially be used as the basis for approximating the solutions of several other flows over time problems.

When multiple sources and sinks are considered for the flows-over-time problem, a straightforward time expanded graph construction is not sufficient. Hoppe and Tardos[6] study the *quickest transshipment problem*, discovering it to be much more challenging than the case with a single source/sink pair. The overall "complexity landscape" of these problems is described in a nice review by Kohler, Mohring, and Skutella[2]. In[8], reference is made to a personal communication by A. Hall and M. Skutella indicating the multicommodity flows over time problem is NP-hard.

Note that virtually all of the approaches above rely on construction of a form of time-expanded graph generated from the original graph. In our work, we instead formulate the problem directly with respect to the given graph as a linear program. As a consequence, however, we must construct a special time interval set that is in some ways analogous to the arcs in a time-expanded graph.

For completeness, it is worth mentioning that several researchers have begun to focus on problems involving flows over time in graphs where arc traversal times are a function of the flows assigned to the arc. Such cases arise in the modelling of congested networks, including highways. These problems can be daunting, given the highly complex arc delay functions thought to exist for

real-world traffic network situations. This problem is more complicated than the one we are exploring here for a variety of reasons, and the interested reader is referred to Carey and Subrahmanian[9] and Kohler, Langkau and Skutella[3].

8 Conclusions

We have presented and algorithm to solve a version of a routing problem in delay tolerant networking in which contact possibilities are known in advance. The algorithm starts with a finite set of contacts with (possibly variable) capacities and delays, and produces an optimal schedule assigning messages to edges and times, that respects message priority and minimizes the overall delivery delay.

There are two steps to the algorithm. A first crucial step is to obtain from the contacts a discretization of time satisfying a certain property (Sections 2, 4 and 5). In the usual treatment of dynamic flow problems in the literature, discrete time plays the role of a "period" or "phase" to be repeated. By contrast, in delay tolerant networking the time points at which to inject flow, as well as for how long, are of the utmost importance.

The second step is to formulate a linear program that captures the characteristics of the problem. In fact, we present two equivalent LP formulations of the problem, of which one is smaller and runs faster in CPLEX, a general purpose linear solver. The linear programs are dynamic versions of the classical multicommodity flow problem, where both edges and nodes are capacitated (in the case of edges, with piece-wise constant capacities). Delay is modelled by distinguishing between transmission and reception: a set of variables is introduced to deal with transmission (at the beginning of a link) and another set to deal with reception (at the endpoint of the link). Variables in the two sets are related by the condition that whatever was transmitted (at the beginning of a link) during a certain time interval, is received (at the end of the link) during the same interval *translated* by the delay the link in question has during the given time interval.

References

[1] K. Fall "A Delay Tolerant Network Architecture for Challenged Internets" Proc. SIGCOMM 2003 Aug., 2003

[2] E. Köhler, R. Möhring, M. Skutella. "Traffic Networks and Flows over Time" TU-Berlin Technical Report 752/2002 2002.

[3] E. Köhler, K. Langkau, M. Skutella. "Time-Expanded Graphs for Flow-Dependent Transit Times" TU-Berlin Technical Report 751/2002 2002.

[4] L.R. Ford, D.R. Fulkerson *Flows in Networks* Princeton University Press Princeton, NJ, 1962

[5] L.R. Ford, D.R. Fulkerson "Constructing maximal dynamic flows from static flows" Operations Research 6:419-433, 1958

[6] B. Hoppe, È. Tardos "The quickest transshipment problem" Proc. 6th ACM-SIAM symp. on Discrete Algorithms (SODA) 1995

[7] L. Fleischer, M. Skutella "The quickest multicommodity flow problem" in Integer Programming and Combinatorial Optimization W.J. Cook, A.S. Schulz, eds Lecture Notes in Computer Science, vol 2337 2002, pp. 36-53

[8] L. Fleischer, M. Skutella "Minimum Cost Flows over Time without Intermediate Storage" Proc. SODA 2003

[9] M. Carey, E. Subrahmanian "An approach to modelling time-varying flows on congested networks" Transportation Research B Volume 34 (2000), pp.157-183